

A Neural Joint Model for Extracting Bacteria and Their Locations

Fei Li¹, Meishan Zhang², Guohong Fu², and Donghong Ji¹(✉)

¹ School of Computer, Wuhan University, Wuhan, China
{lifei_csnlp,dhji}@whu.edu.cn

² School of Computer Science and Technology, Heilongjiang University,
Harbin, China

Abstract. Extracting *Lives.In* relations between bacteria and their locations involves two steps, namely bacteria/location entity recognition and *Lives.In* relation classification. Previous work solved this task by pipeline models, which may suffer error propagation and cannot utilize the interactions between these steps. We follow the line of work using joint models, which perform two subtasks simultaneously to obtain better performances. A state-of-the-art neural joint model for relation extraction in the Automatic Content Extraction (ACE) task is adapted to our task. Furthermore, we propose two strategies to improve this model. First, a novel relation is suggested in the second step to detect the errors in the first step, thus this relation can correct some errors in the first step. Second, we replace the original greedy-search decoding with beam-search, and train the model with early-update techniques. Experimental results on a standard dataset for this task show that our adapted model achieves better precisions than other systems. After adding the novel relation, we gain a nearly 2% improvement of F1 for *Lives.In* relation extraction. When beam-search is used, the F1 is further improved by 6%. These demonstrate that our proposed strategies are effective for this task. However, additional experiments show that the performance improvement in another dataset of bacteria and location extraction is not significant. Therefore, whether our methods are effective for other relation extraction tasks needs to be further investigated.

Keywords: Bacteria · Biotope · Relation extraction · Joint model

1 Introduction

The information of bacteria and their surviving environments is useful in many areas such as food safety and health sciences. Therefore, extracting bacteria and their locations has received much research attention in the biomedical natural language processing (BioNLP) community [2, 6, 14]. Taking a sentence “The vibrios are ubiquitous to oceans.” in the guideline of the Bacteria Biotope (BB) task at BioNLP shared task (BioNLP-ST) 2016 [3] as an example, the task aims to extract bacteria entity mentions (e.g., vibrios), location entity mentions (e.g., oceans), and *Lives.In* relations (e.g., {vibrios, oceans}) from this sentence.

This is a typical relation extraction task that involves two steps. First, entity mentions are recognized and second, each pair of entity mentions is examined, deciding whether a *Lives_In* relation exists. The first step can be treated as a named entity recognition (NER) task [7], and the second step can be casted as a relation classification task [19]. We focus on the line of work using neural networks, which have achieved state-of-the-art performances for both tasks.

Recently, Miwa and Bansal [13] proposed a neural joint model for relation extraction in the ACE task¹, which can be adapted to our task. Compared with pipeline models that handle NER and relation classification separately, joint models can alleviate the problem of error propagation [9]. For example, if the bacteria or location entity of a *Lives_In* relation is not correctly recognized, this relation will be definitely lost. Another advantage of joint models is that they can utilize the interactions between two steps. Miwa and Bansal [13] implicitly performed it by building the features of the second task based on the outputs of the first task, and jointly training these features. To enhance the interactions explicitly, we add a special relation called *Invalid_Entity*, which means that some entities related to such relation may be incorrectly recognized. If an entity is only associated with *Invalid_Entity* relations, it will be removed from final results of entity recognition. Thus, even if there are some wrongly-recognized entities, we can still correct them by the second step.

Moreover, Miwa and Bansal [13] exploited a greedy left-to-right manner to predict entity recognition labels incrementally, which may suffer error propagation among these labels, i.e., the error in the prior prediction can induce new errors in the subsequent predictions. In this paper, we use beam-search, which has been successfully applied in other tasks [9, 21], to alleviate this problem.

We adapt the model of Miwa and Bansal [13] as our baseline, and verify our strategies gradually in the BB task at BioNLP-ST 2016 [3], which is a standard competition for *Lives_In* relation extraction between bacteria and location entities. Results show that our baseline can achieve state-of-the-art performances for this task. By adding the *Invalid_Entity* relation, we gain a nearly 2% improvement of F1. When beam-search is used, the F1 is further improved by 6%.

2 Related Work

Extracting *Lives_In* relations between bacteria and location entities belongs to the line of work on relation extraction. Prior work usually used two-step pipeline models to handle this task [2, 4, 14]. First, all possible bacteria/location entities are recognized using sequence labeling models. Then *Lives_In* relations are extracted between bacteria/location entity pairs using binary classification models. We do not exploit this framework because it can easily suffer the error propagation problem. Moreover, the useful interaction information between two steps is unable to be incorporated.

Our work falls into the line of work using joint models for relation extraction. Roth and Yih [16] proposed a joint inference framework based on integer

¹ <https://www ldc.upenn.edu/collaborations/past-projects/ace>.

linear programming to extract entities and relations. Li and Ji [9] exploited a single transition-based model to accomplish entity recognition and relation classification simultaneously. Kordjamshidi et al. [6] proposed a structured learning model to extract biomedical entities and their relationships. Very recently, Miwa and Bansal [13] proposed a neural model based on long short-term memories (LSTMs) [5] to perform relation extraction jointly. This model captures both word sequence and dependency structure information by stacking tree-structured recurrent neural networks (RNNs) on sequential RNNs, which allows the model to share parameters between two submodules of entity recognition and relation classification. Such method utilizes the correlations between the relevant sub-tasks for mutual benefit, and outperforms state-of-the-art feature-based model [9, 16]. We follow the work of Miwa and Bansal [13], with extensions of a novel interaction mechanism and beam-search [9, 21].

Our work is also related to neural network models of NER [7], relation classification [8, 12, 17–19] and relation extraction [10]. For NER, Lample et al. [7] exploited RNNs to extract features, which are similar with our neural network structures for NER. For relation classification, Zeng et al. [19] leveraged convolutional neural networks (CNNs) to classify relations with lexical, sentence and word position features. Li et al. [8] used the similar framework and features, but focused on *Lives_In* relation classification between bacteria and their locations. In particular, our neural network structures of relation classification are similar with [12, 18], which exploited RNNs over the shortest dependency path between two target entities to extract neural features. For relation extraction, prior work focused on distant supervised methods using Freebase [10], whose methods and tasks are essentially different from ours.

3 Baseline

We follow the work of Miwa and Bansal [13] to build our baseline for extracting bacteria and their locations. Figure 1 shows an example of the analysis process when a sentence “The vibrios are ubiquitous to oceans.” is given.

3.1 Bacteria/Location Entity Recognition

The model casts bacteria/location entity recognition as a sequence labeling problem. The output sequence labels are defined to recognize three entity types in our task with a *BILOU* scheme [7], where *B-Bacteria*/*B-Habitat*/*B-Geographical*, *I-Bacteria*/*I-Habitat*/*I-Geographical* and *L-Bacteria*/*L-Habitat*/*L-Geographical* denote the beginning, following and last words of bacteria/habitat/geographical entities. *U-Bacteria*/*U-Habitat*/*U-Geographical* denote the only words of corresponding entities, and *O* denotes that the word does not belong to any type of entities. Following the task definition [3], we consider that both habitat and geographical entities are location entities.

Our model predicts the entity label of each word from left to right. Given an input sentence $w_1/t_1, w_2/t_2, \dots, w_n/t_n$, where w denotes a word and t denotes

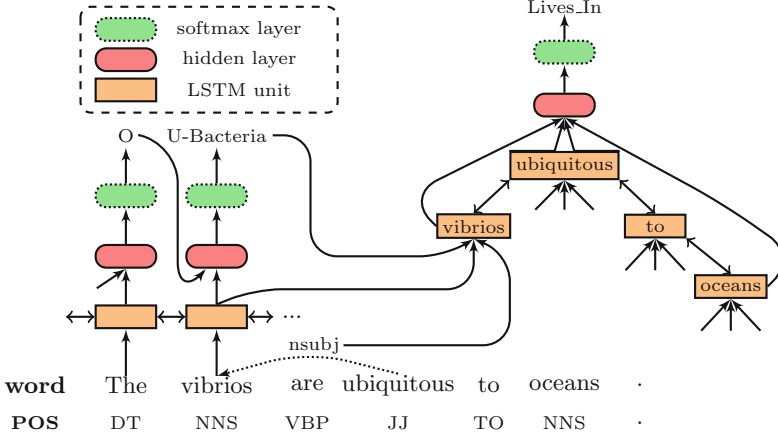


Fig. 1. “vibrios” and “oceans” are bacteria and location entities. “POS” denotes part-of-speech tags. The dotted arrow line denotes a “nsubj” dependency type between “vibrios” and its governor “ubiquitous”. “O” and “U-Bacteria” denote entity labels, and “Lives_In” denotes a relation label. The left part recognizes bacteria/location entities by tagging each word with an entity label from left to right incrementally. The right part determines whether a *Lives_In* relation exists between a pair of bacteria/location entities by building a dependency tree and extracting features from it.

its POS tag. We represent each w_i/t_i by concatenating their embeddings, namely $x_i = [e(w_i); e(t_i)]$. A bi-directional LSTM-RNN is built based on x_1, x_2, \dots, x_n , and outputs h_1, h_2, \dots, h_n . h_i is selected as one source of features to predict the entity label l_i of w_i/t_i . The label l_{i-1} of last word is selected as another source. Finally, we concatenate h_i and $e(l_{i-1})$, and use a feed-forward neural network with a hidden layer s_i and a softmax layer to compute the scores of all entity labels. The label with the highest score is selected as l_i for w_i/t_i .

3.2 *Lives_In* Relation Extraction

Once entity recognition is finished, we start binary relation classification to determine whether a *Lives_In* relation exists between a pair of bacteria and location entities. The key idea of the classification is to build a dependency tree whose root is the lowest common ancestor of two target entities, and model the shortest dependency path between the ancestor and target entities.

As shown in the right part of Fig. 1, given two target entities a (e.g., vibrios), b (e.g., oceans) and their lowest common ancestor c (e.g., ubiquitous) in the dependency tree. The shortest dependency paths can be formally represented by $\{a, a_1, \dots, a_m, c, b_n, \dots, b_1, b\}$ (e.g., $\{\text{vibrios}, \text{ubiquitous}, \text{to}, \text{oceans}\}$), where a_1, \dots, a_m or b_1, \dots, b_n denotes the words occurred on the path between a and c , or b and c , respectively. The path can be divided into two parts, where $\uparrow \text{seq}_a = \{a, a_1, \dots, a_m, c\}$ (e.g., $\{\text{vibrios}, \text{ubiquitous}\}$) and $\uparrow \text{seq}_b = \{b, b_1, \dots, b_n, c\}$ (e.g., $\{\text{oceans}, \text{to}, \text{ubiquitous}\}$) are bottom-up sequences, and $\downarrow \text{seq}_a =$

$\{c, a_m, \dots, a_1, a\}$ (e.g., {ubiquitous, vibrios}) and $\downarrow seq_b = \{c, b_n, \dots, b_1, b\}$ (e.g., {ubiquitous, to, oceans}) are top-down sequences.

Features are extracted from these sequences by LSTMs. The input of each LSTM unit is a concatenation of three parts, $x_i = [h_i; e(l_i); e(d_i)]$, where h_i is the output of the LSTM unit for entity recognition in Sect. 3.1. $e(l_i)$ and $e(d_i)$ are the entity label and dependency type embeddings. The last outputs of LSTMs computing along $\uparrow seq_a, \uparrow seq_b, \downarrow seq_a$ and $\downarrow seq_b$ are $\uparrow h_a, \uparrow h_b, \downarrow h_a$ and $\downarrow h_b$. Finally, $\uparrow h_a, \uparrow h_b, \downarrow h_a$ and $\downarrow h_b$ are fed into a hidden layer s_{ab} , and a softmax layer is used to compute the scores of all relation labels. The label with the highest score is selected as the relation type of target entities.

3.3 Training

Both parts of the neural network in Fig. 1 employ the same training algorithm based on stochastic gradient decent, so we describe their training in one section for conciseness. The final training objective based on cross-entropy losses is

$$L(\theta) = -\sum_i \log p_y + \frac{\lambda}{2} \|\theta\|_2^2, \quad (1)$$

where θ denotes all the model parameters, y denotes the gold label of a training example, p_y denotes the probability predicted by our model, and λ denotes the regularization parameter of L_2 regularization term. We exploit back propagation to compute the gradients of model parameters.

4 Our Method

4.1 Invalid_Entity Relation

In our baseline, the two subtasks, entity recognition and *Lives_In* relation extraction, have their own neural network structures, respectively. The two sub-networks share several common inputs, thus the two subtasks are mutually affected. In addition, the training losses of relation classification network can be propagated back into the entity recognition network. All these interactions are performed implicitly through the sharing of model parameters, because parameter weights of both sub-networks are influenced by losses of both subtasks.

However, we aim to make the upper relation classification task help the entity recognition task explicitly. In the baseline model, the relation classification submodule handles two categories, namely *Lives_In* relation and not *Lives_In* relation. It is built upon the assumption that the given entity pair is a real bacteria/location pair, which cannot be corrected when the entity recognition submodule makes errors. In order to handle this case, we add a relation *Invalid_Entity* to the relation classification submodule. This relation indicates that at least one of two target entities recognized in the first step is incorrect. If an entity is only associated with *Invalid_Entity* relations, it will be removed from final results

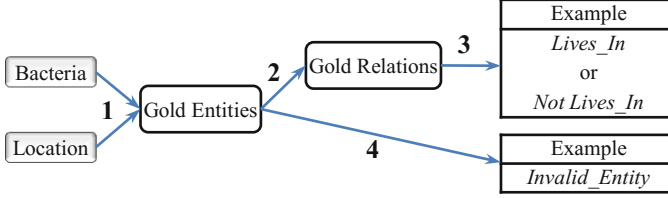


Fig. 2. Training procedure of relation classification. 1: Given a pair of recognized bacteria and location entities, we match them with gold entities. 2: If both of them can be matched, we search their gold relation type in gold relations by entities. 3: A training example is built with the gold relation type, namely *Lives_In* or not *Lives_In*. 4: If any of them cannot be matched with gold entities, this entity pair is impossible to be associated with any gold relation. Therefore, a training example is built with the *Invalid_Entity* relation type.

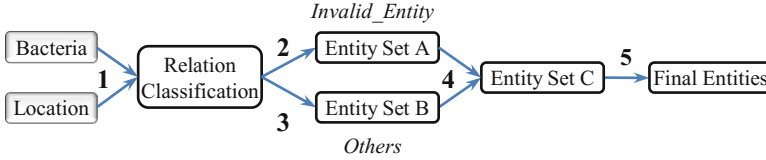


Fig. 3. Decoding procedure of relation classification. 1: Given a pair of recognized bacteria and location entities, our model predicts their relation. 2: If the predicted relation is *Invalid_Entity*, we add two target entities into the entity set A. 3: If the predicted relation is others (either *Lives_In* or not *Lives_In*), we add two target entities into the entity set B. 4: The entity set C denotes the set difference (A-B) of set A and B, so the entities in the set C are only associated with *Invalid_Entity* relations. 5: Entities in the set C will be removed from the final results of entity recognition, and the relations related to the entities in the set B can be used as the final results of relation classification.

of entity recognition. This relation can further help us to correct several errors made by the entity recognition submodule.

After adding the *Invalid_Entity* relation, the training procedure of entity recognition does not change but that of relation classification changes as shown in Fig. 2. Similarly, the decoding procedure of relation classification changes correspondingly as shown in Fig. 3.

4.2 Beam-Search

During entity recognition, our baseline model exploited a greedy left-to-right manner to assign an entity label to each word. The prediction of next step requires the entity label of current step. Thus, when the current step is incorrect, it could influence the result of the next step. This kind of error propagation is less severe than that of pipeline models, because the parameters of joint models are trained jointly and the errors could be considered implicitly to some extent.

```

1: agenda  $\leftarrow \{ \}$ 
2: for word in sentence
3:   beam  $\leftarrow \{ \}$ 
4:   for candidate in agenda
5:     for label in entity labels
6:       score  $\leftarrow \text{COMPUTE}(\text{candidate}, \text{word}, \text{label})$ 
7:       beam  $\leftarrow \text{NEWCANDIDATE}(\text{candidate}, \text{word}, \text{label}, \text{score})$ 
8:   agenda  $\leftarrow \text{TOPK}(\text{beam})$ 
9: best  $\leftarrow \text{BEST}(\text{agenda})$ 
10: entities  $\leftarrow \text{CREATEENTITY}(\text{best})$ 

```

Fig. 4. Beam-search decoding for entity recognition.

For each word (i.e., step) in a given sentence, we firstly fetch a history candidate prediction in *agenda* (line 4), and then give a score for each entity label based on the candidate and the current word (line 6). After that, a new candidate prediction is generated and added to *beam* (line 7). After all the candidates in *agenda* have been iterated, we rank the candidates in *beam* (line 8) by accumulating the entity label score of each word in each candidate, formally by

$$\text{score}(\text{candidate}) = \sum_{l_i \in L} \text{score}(l_i) = \sum_{l_i \in L} w \cdot f(l_i), \quad (2)$$

where $L = \{l_1, l_2, \dots, l_n\}$ denotes the entity label sequence of the current candidate, w denotes the model parameters and f denotes the feature extraction function. K-best candidates are stored back into *agenda* for next step (line 8). After the last step, we use the best candidate prediction and create entities based on it (lines 9–10). The advantage of beam-search is that we have multiple choices at each step, in case that the optimal local prediction is incorrect. The candidate predictions are ranked by global scores, thus error propagation can be alleviated.

We exploit the early-update strategy [9, 21] during training, which has been widely used with beam-search. The updating of model parameters is performed at the time when gold-standard results cannot be recovered by the predicted candidates in the beam. Thus only the losses of partial results are used for back propagation. In Fig. 4, the early-update strategy is applied immediately after fetching the k-best candidates in the beam at each step (line 8).

5 Experiments

5.1 Experimental Settings

We conduct experiments on a standard dataset from the BB task at BioNLP-ST 2016 [3], which includes an open competition named *BB-event+ner*. In this competition, gold entities are not given, so participants need to perform both bacteria/location entity recognition and *Lives_In* relation extraction. The dataset consists of 161 documents from PubMed, and we follow the official method to split

Table 1. Hyper-parameter settings, where D denotes vector dimensions.

Type	Hyper-parameter
Training	$\alpha = 0.01, \lambda = 10^{-8}$
Embedding	$D(e(w_i)) = 200$
	$D(e(t_i)), D(e(d_i))$ or $D(e(l_i)) = 25$
Entity Network Structure	$D(h_i) = 200, D(s_i) = 100$
Relation Network Structure	$D(\uparrow h_a, \uparrow h_b, \downarrow h_a \text{ or } \downarrow h_b) = 100$
	$D(s_{ab}) = 100$

the dataset into training, development and test sets. In particular, we remove the discontinuous and nested entities, in order to fit our models.

For the development set, we use precision (P), recall (R) and F1-score (F1) to evaluate the performances of entity recognition and relation extraction. A recognized entity is counted as true positive (TP) if its boundary and type match those of a gold entity. An extracted relation is counted as TP if its relation type is correct, and the boundaries and types of its related entities match those of the entities in a gold relation. For the test set, we use the official evaluation service², which shows only the overall performance (P, R, F1) of relation extraction.

Hyper-parameters are tuned based on the development set. In Table 1, α and λ denotes the learning rate and L_2 regularization parameter. “Entity Network Structure” and “Relation Network Structure” denote the structures of neural networks for entity recognition and relation classification, respectively. “Embedding” denotes the basic features we used. We use pre-trained biomedical word embeddings [15] to initial our word embeddings and other kinds of embeddings are randomly initialized in the range $(-0.01, 0.01)$.

Given a document, we split it into sentences and then tokenize these sentences. All the tokens are transformed into lowercase forms and numbers are replaced by zeroes. Stanford CoreNLP toolkit [11] is used for POS tagging and dependency parsing. Neural networks are implemented based on LibN3L [20].

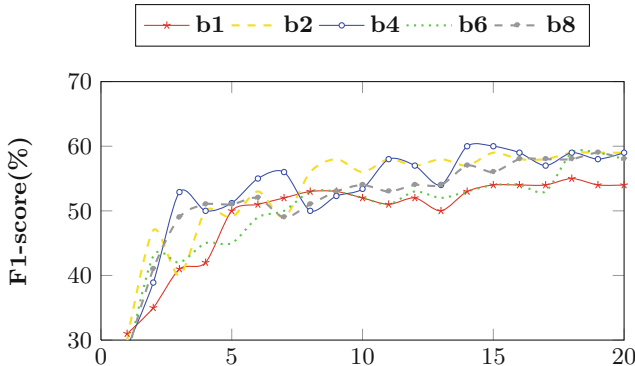
5.2 Development Results

As shown in Table 2, our model improves F1 in bacteria/location entity recognition by 0.6% after adding the *Invalid_Entity* relation. The performance improvement is mainly due to the growth of precision. A likely reason may be that some incorrectly-recognized entities are removed. This demonstrates that the relation classification submodule can help the entity recognition submodule to correct some errors through the *Invalid_Entity* relation. In addition, the F1-score of *Lives_In* relation extraction also increases, from 16.3% to 20.3%. It demonstrates that the *Invalid_Entity* relation can help to boost relation extraction as well. The

² <http://bibliome.jouy.inra.fr/demo/BioNLP-ST-2016-Evaluation/index.html>.

Table 2. Developmental results (%) of the baseline and our proposed methods.

Method	Entity recognition			Relation extraction		
	P	R	F1	P	R	F1
Baseline	63.6	47.9	54.7	24.2	12.3	16.3
+Invalid_Entity	68.8	46.2	55.3	25.0	17.2	20.3
+Beam (4)	69.7	51.8	59.4	27.8	20.9	23.9

**Fig. 5.** F1 against the training epoch using different beam sizes for entity recognition.

possible reason is that this relation can divide the non-*Lives_In* relations more reasonably. Overall, the results demonstrate that *Invalid_Entity* can boost the performances of both entity recognition and relation extraction.

Figure 5 shows the development results of beam-search, namely the F1 scores of entity recognition with respect to the training epoches. We experiment with five beam settings, including beam 1, 2, 4, 6 and 8, where beam 1 denotes the baseline greedy search. With beam-search (the beam size is larger than 1), the performance of entity recognition outperforms the baseline method. According to Fig. 5, we set the final beam size by 4, which achieves the best performance. In Table 2, we also show the concrete developmental results of both bacteria/location entity recognition and *Lives_In* relation extraction. The recall values of relation extraction are greatly boosted by beam-search, which is similar with our *Invalid_Entity* strategy. Actually, we do not use beam-search in the relation classification phase, thus the main benefit comes from entity recognition, which brings better performances for the overall relation extraction as well. Overall, beam-search can give a further increase of 3.6% in F1 for relation extraction.

5.3 Final Results

Table 3 shows the final overall relation extraction results of our models. The baseline model can obtain 20.7% of F1, and after adding the *Invalid_Entity*

Table 3. Final results (%) on the test set.

Method	Relation extraction		
	P	R	F1
Our models			
Baseline	46.9	13.2	20.7
+Invalid_Entity	46.1	14.9	22.5
+Beam	46.1	20.7	28.5
Other work			
LIMSI	19.3	19.1	19.2
UTS	33.1	13.3	19.0

Table 4. Developmental results (%) of the BB 2013 task.

Method	Entity recognition			Relation extraction		
	P	R	F1	P	R	F1
Baseline	79.3	74.0	76.6	36.3	6.5	11.0
+Invalid_Entity	79.9	74.3	77.0	37.3	6.8	11.5
+Beam	81.7	76.2	78.9	28.5	8.1	12.7

relation, F1 is boosted by 1.8%. When beam-search is applied, we can have a further improvement of 6%, which demonstrates our proposed strategies are useful. In particular, we find our strategies can mainly contribute to the recall values, which is consistent with the finding on the development set. Considering the low proportion of *Lives_In* relations, the recall is highly important.

Moreover, we show the performances of the top-two systems for this task, namely LIMSI and UTS, which both leverage pipeline models. LIMSI [4] uses conditional random field (CRF) and post-processing rules to identify mentions of bacteria and locations, and support vector machine (SVM) to classify *Lives_In* relations between two entity mentions. UTS [3] relies on two independent SVMs to perform entity recognition and relation classification, respectively. From Table 3, we can see that they suffer either lower precision or recall.

5.4 Additional Experiments

We also additionally evaluated our method on the subtask 3 of the bacteria biotope (BB) task [1] in the BioNLP 2013 shared task. The BB 2013 task is similar with the BB 2016 task [3], and we focused on the extraction of *Localization* relations which represent the same meaning as *Lives_In* relations. The BB 2013 task includes 78, 27 and 26 documents as training, development and test sets. Since the official evaluation service is unavailable, we used the development set for evaluation. The experimental settings of the BB 2013 task is identical to those of the BB 2016 task, and the development results are shown in Table 4.

If the *Invalid_Entity* relation is added, the precision, recall and F1-score of entity recognition increase slightly (0.6%, 0.3% and 0.4%), and those of relation extraction also rise by 1.0%, 0.3% and 0.5% respectively. By utilizing beam-search, the precision, recall and F1 of entity recognition further increase by 1.8%, 1.9% and 1.9%, and the performance of relation extraction is generally improved except the precision, declining by 8.8%. Overall, the performance improvement in the BB 2013 task is not as apparent as that in the BB 2016 task.

6 Conclusion

To extract bacteria and their habitats, we employed a state-of-the-art system for joint entity and relation extraction. To enhance this system, two extensions were made. First, we added the *Invalid_Entity* relation to model the conditions with incorrectly recognized bacteria/location entities. Then we applied beam-search to replace the greedy decoding. Experimental results on a benchmark dataset showed that both of our extensions could improve the performance significantly. We demonstrate that implicit parameter sharing for joint models is not enough and greedy decoding also influences the performance of joint models. However, additional experiments on another dataset showed that the performance improvements were not obvious. Therefore, we need to evaluate our method on more relation extraction tasks to further demonstrate its effectiveness.

Acknowledgments. This work is supported by the National Natural Science Foundation of China (No. 61373108), the National Philosophy Social Science Major Bidding Project of China (No. 11&ZD189). This work is also supported by Humanities and Social Science Foundation of Ministry of Education of China (16YJCZH004), and the China Postdoctoral Science Foundation (2014T70722).

References

1. Bossy, R., Golik, W., Ratkovic, Z., Valsamou, D., Bessi eres, P., N edellec, C.: Overview of the gene regulation network and the bacteria biotope tasks in BioNLP 2013 shared task. *BMC Bioinform.* **16**(10), S1 (2015)
2. Claveau, V.: IRISA participation to BioNLP-ST13: lazy-learning and information retrieval for information extraction tasks. In: *Proceedings of the BioNLP Shared Task 2013 Workshop* (2013)
3. Del eger, L., Bossy, R., Chaix, E., Ba, M., Ferr e, A., Bessi eres, P., N edellec, C.: Overview of the bacteria biotope task at BioNLP shared task 2016. In: *Proceedings of the 4th BioNLP Shared Task Workshop* (2016)
4. Grouin, C.: Identification of mentions and relations between bacteria and biotope from PubMed abstracts. In: *Proceedings of the 4th BioNLP Shared Task Workshop* (2016)
5. Hochreiter, S., Schmidhuber, J.: Long short-term memory. *Neural Comput.* **9**(8), 1735–1780 (1997)
6. Kordjamshidi, P., Roth, D., Moens, M.F.: Structured learning for spatial information extraction from biomedical text: bacteria biotopes. *BMC Bioinform.* **16**, 129 (2015)

7. Lample, G., Ballesteros, M., Subramanian, S., Kawakami, K., Dyer, C.: Neural architectures for named entity recognition. In: *Proceedings of the 2016 Conference of the NAACL*, pp. 260–270 (2016)
8. Li, H., Zhang, J., Wang, J., Lin, H., Yang, Z.: DUTIR in BioNLP-ST 2016: utilizing convolutional network and distributed representation to extract complicate relations. In: *Proceedings of the 4th BioNLP Shared Task Workshop* (2016)
9. Li, Q., Ji, H.: Incremental joint extraction of entity mentions and relations. In: *Proceedings of the 52nd ACL*, pp. 402–412 (2014)
10. Lin, Y., Shen, S., Liu, Z., Luan, H., Sun, M.: Neural relation extraction with selective attention over instances. In: *Proceedings of the 54th Annual Meeting of the ACL*, pp. 2124–2133 (2016)
11. Manning, C.D., Surdeanu, M., Bauer, J., Finkel, J., Bethard, S.J., McClosky, D.: The Stanford coreNLP natural language processing toolkit. In: *Proceedings of 52nd ACL*, pp. 55–60, September 2014
12. Mehryary, F., Björne, J., Pyysalo, S., Salakoski, T., Ginter, F.: Deep learning with minimal training data: TurkuNLP entry in the BioNLP shared task 2016. In: *Proceedings of the 4th BioNLP Shared Task Workshop* (2016)
13. Miwa, M., Bansal, M.: End-to-end relation extraction using LSTMs on sequences and tree structures. In: *Proceedings of the 54th Annual Meeting of the ACL*, pp. 1105–1116 (2016)
14. Nguyen, N., Tsuruoka, Y.: Extracting bacteria biotopes with semi-supervised named entity recognition and coreference resolution. In: *Proceedings of the BioNLP Shared Task 2011 Workshop*, pp. 94–101 (2011)
15. Pyysalo, S., Ginter, F., Moen, H., Salakoski, T., Ananiadou, S.: Distributional semantics resources for biomedical text processing. In: *LBM* (2013)
16. Roth, D., Yih, W.: Global inference for entity and relation identification via a linear programming formulation. In: *Introduction to Statistical Relational Learning* (2007)
17. Wang, L., Cao, Z., de Melo, G., Liu, Z.: Relation classification via multi-level attention CNNs. In: *Proceedings of the 54th Annual Meeting of the ACL*, pp. 1298–1307 (2016)
18. Xu, Y., Mou, L., Li, G., Chen, Y., Peng, H., Jin, Z.: Classifying relations via long short term memory networks along shortest dependency paths. In: *Proceedings of the EMNLP*, pp. 1785–1794 (2015)
19. Zeng, D., Liu, K., Lai, S., Zhou, G., Zhao, J.: Relation classification via convolutional deep neural network. In: *Proceedings of the 25th COLING*, pp. 2335–2344 (2014)
20. Zhang, M., Yang, J., Teng, Z., Zhang, Y.: LibN3L: a lightweight package for neural NLP. In: *Proceedings of the Tenth International Conference on Language Resources and Evaluation*, pp. 23–28 (2016)
21. Zhang, M., Zhang, Y., Fu, G.: Transition-based neural word segmentation. In: *Proceedings of the 54th Annual Meeting of the ACL*, pp. 421–431 (2016)

Advances in Knowledge Discovery and Data Mining
21st Pacific-Asia Conference, PAKDD 2017, Jeju, South
Korea, May 23-26, 2017, Proceedings, Part II

Kim, J.; Shim, K.; Cao, L.; Lee, J.-G.; Lin, X.; Moon, Y.-S.
(Eds.)

2017, XXXII, 857 p. 252 illus., Softcover

ISBN: 978-3-319-57528-5